TITLE OF THE INVENTION

Normalized bitmap representation of Visual Object's Shape for Search/Query/Filtering Applications

CROSS REFERENCE TO RELATED APPLICATIONS

This is a continuation of provisional U.S. Patent Application Serial No. 60/118,207 filed February 1, 1999, now abandoned.

BACKGROUND OF THE INVENTION

The present invention relates to video data processing, and more particularly to a normalized bitmap representation of a visual object's shape for search/query/filtering applications.

With the success of the Internet, and picture and video coding standards such as JPEG, MPEG-1, 2, more and more audio-visual information is available in digital form. Before one can use any such information, however, it first has to be located. Searching for textual information is an established technology. Many text-based search engines are available on the World Wide Web to search text documents. Searching is not yet possible for audio-visual content, since no generally recognized description of this material exists. MPEG-7 is intended to standardize the description of such content. This description is intended to be useful in performing search at a very high level or at a low level. At a high level the search may be to locate "a person wearing a white shirt walking behind a person wearing red sweater". At lower levels for still images the search may use characteristics like color, texture and information about the shape of objects in the picture. The high level queries may be mapped to the low-level primitive queries to perform the search.

Visual object searches are useful in content creation, such as to locate from archive the footage from a particular event, e.g. a tanker on fire, clips containing particular public figure, etc. Also the number of digital broadcast channels is increasing every day. One search/filtering application is to be able to select the broadcast channel (radio or TV) that is potentially interesting.

What is desired is a descriptor that may be automatically or semi-automatically extracted from still images/key images of video and be used in searches.

BRIEF SUMMARY OF THE INVENTION

Accordingly the present invention provides a normalized bitmap representation of a visual object's shape for search/query/filtering applications that is easy to compute, but answers a variety of queries. An image is segmented into visual objects, and the samples belonging to one of the visual objects of interest are identified and grouped into the largest connected blob as an un-normalized bitmap. The un-normalized bitmap is normalized with respect to translation, rotation and scale by estimating the mean and covariance of the samples and back projecting the un-normalized bitmap as a function of the mean and a principal direction to produce a normalized bitmap representation having a standard height and oriented so the principal direction is along a vertical direction. Once all visual objects in an image database have associated normalized bitmap representations, a query bitmap may be used to identify those visual objects from the database that have a desired shape, aspect ratio or sample density.

The objects, advantages and other novel features of the present invention are apparent from the following detailed description when read in conjunction with the appended claims and attached drawing.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWING

Fig. 1 is an illustrative view of a visual object with its binding box according to the present invention.

Fig. 2 is an illustrative view of a bitmap representing the object shape of Fig. 1.

Fig. 3 is an illustrative view of an unnormalized bitmap.

Fig. 4 is an illustrative view of a normalized bitmap according to the present invention.

Fig. 5 is a block diagram view of an overall normalized bitmap generation according to the present invention.

Fig. 6 is a block diagram view of a bitmap normalization process according to the present invention.

Fig. 7 is a block diagram view of a search engine based on normalized bitmap representation of shape according to the present invention.

Fig. 8 is an illustrative view of a mismatch measure according to the present invention.

Fig. 9 is an illustrative view of the generation of four versions of a query shape according to the present invention.

Fig. 10 is an illustrative view of the results from a query for similar shapes according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

A normalized bitmap representation of a visual object's shape may be used for searching based on the shape of the object. This representation is easy to compute, but answers a variety of queries that are described later. This representation provides a high quality shape representation. However, this representation has somewhat larger memory requirements compared to the simple binding box representation, described co-pending provisional U.S. Patent Application Serial No. 60/118,386 entitled "Coarse representation of visual object's shape for search/query/filtering applications". Loss-less bitmap compression methods, such as arithmetic coding, may be used to minimize the memory requirements. Use of lossy compression methods results in further reductions in memory requirements, with very little loss in search performance. The bitmap representation is more complete than the contour-based representation of the visual object shape, in the sense that it can easily account for "holes" in the shape.

This method may be used for visual objects in still images or in video. A bitmap is a matrix of numbers, as shown in Fig. 2, whose dimensions are at least that of a binding box encompassing the visual object of interest in an image, and may go up to the dimensions of the image itself. A value of "1" at a position in this bitmap is an indication that the particular pixel belongs to the object. The binding box of the visual object is the tightest rectangle that fully encompasses that visual object in the image, as shown in Fig. 1. In general each semantic object, or its sub-portions, may be represented by a bitmap whose values are defined inside a binding box (a rectangle) in the image.

A generic bitmap representation of the shape is not a suitable format for matching needed in servicing queries. For example, the resolution of a video capture device dictates the number of samples of the bitmap that represent the object. If a standard definition camera captures a given tree, the number of samples contained in the bitmap is lower than what would be contained if an HDTV camera were used. In addition, the orientation of the camera, the zoom parameters and the spatial positioning of the tree within the picture affect the bitmap. In all these situations, the visual object is the same, and a search for objects of that shape is desired. In order to neutralize all these capture-time variables, a normalized bitmap representation is used for shape, which more easily matches with a query shape.

The normalized bitmap representation of the shape enables resolution, orientation, location, and flip agnostic match of two shapes. The following describes the normalization process for the bitmap and the matching process to determine the "closeness" or "mismatch" between two given shapes. This mismatch measure may be used to extract best matches from a database of visual object shapes. Fig. 3 shows an unnormalized bitmap. Figure 4 shows the corresponding normalized bitmap.

The steps involved in the generation of a normalized bitmap are (1) segmentation, (2) extraction of un-normalized bitmap, and finally (3) normalization of the bitmap. These stages are cascaded, as shown in Figure 5. In this figure, the segmentation process may either be automatic, semi-automatic, or manual. The segmentation map consists of segmentation labels at each pixel. The set of pixels having a particular segmentation label belongs to a distinct visual object. Thus, the second stage merely creates a binary map, with values "valid" (true, 1, or 255) wherever segmentation label equals the objectID of interest, and "invalid" (false, or 0) elsewhere. Identification of the largest connected region in the bitmap is covered in co-pending provisional U.S. Patent Applications Serial Nos. 60/118,192 and 60/118,208 covering extraction of homeneous regions based on texture or color.

The bitmap normalization is performed in two stages, as shown in Fig. 6. In the estimation of translation and rotation, the mean and covariance of the "positions" of valid samples in the un-normalized bitmap are estimated. The following psuedo code segment illustrates this process.

```
// To figure out translation and rotation parameters
double tmph, tmpv, tmp, sum;
```

```
// Mean is initialized with dimension 2, values 0
Vector<double> mean(2);
// Covariance is initialized with dimension 2x2, values 0
Matrix<double> covar(2, 2);

/* Find mean and covariance of this un-normalized
  bitmap valid coordinates.
*/
for (i = 0; i < inputBitmapHeight; i++) {
   for (j = 0; j < inputBitmapWidth; j++) {
     if (inputBitmap[i][j] is valid) {
               nSamples++;
               tmph = double(j);
               tmpv = double(i);
               mean[0] += tmpv;
               mean[1] += tmph;
               covar[0][0] += (tmpv*tmpv);
               covar[1][1] += (tmph*tmph);
               covar[0][1] += (tmph*tmpv);
     }
   }
}
if (nSamples < 1) {
    Input bitmap is completely invalid;
    Early exit;
}
covar[1][0] = covar[0][1];
mean = (1.0/nSamples) * mean;
covar = (1.0/nSamples) * covar;
covar = covar - mean * Transpose(mean);
```

Based on the mean and covariance estimated above, the principal direction of the input bitmap is computed through Karhunen-Loéve transformation (KLT) of the covariance matrix ("Numerical recipes in C," Press, Teukolsky, Vetterling, and Flannery, Cambridge University Press, 1992). These directions are available as the eigenvectors of the covariance matrix. Once the eigenvectors of the covariance matrix are computed, they are sorted such that the first column of the eigenvector matrix is the dominant eigenvector.

A back-projection process takes as input the mean and eigenvectors computed from the preceding process, and performs the actual normalization of the input bitmap for translation, rotation and scale. After normalization, all bitmaps have a standard height (normalizedHeight), and the bitmaps are oriented such that the principal direction (corresponding to the dominant eigenvector) is along the vertical direction. There are two stages to this back-projection process.

The height of all normalized bitmaps (normalizedHeight) in the database is fixed to a pre-determined quantity. Another way to perform this normalization is to fix the width of all normalized bitmaps. In this implementation, the height is normalized to 65 rows. The width of the normalized bitmap is determined to maintain the aspect ratio identical to the source (i.e. un-normalized bitmap). This is achieved through the following psuedo-code segment, where the bounding box width and height is computed along the new co-ordinate axes.

```
// determine the dimensions upon normalization
tmp = double(inputBitmapWidth+inputBitmapHeight);
double minNewVer = tmp;
double maxNewVer = -tmp;
double minNewHor = tmp;
double maxNewHor = -tmp;
for (i = 0; i < inputBitmapHeight; i++) {
   for (j = 0; j < inputBitmapWidth; j++) {
     if (inputBitmap[i][j] is valid) {
              tmp = (i - mean[0]) * eigvec[0][0] +
                     (j - mean[1]) * eigvec[1][0];
              minNewVer = (minNewVer > tmp) ? tmp : minNewVer;
              maxNewVer = (maxNewVer < tmp) ? tmp : maxNewVer;

              tmp = (i - mean[0]) * eigvec[0][1] +
                     (j - mean[1]) * eigvec[1][1];
              minNewHor = (minNewHor > tmp) ? tmp : minNewHor;
              maxNewHor = (maxNewHor < tmp) ? tmp : maxNewHor;
```

```
    }
        }
}
i = ⌈(maxNewVer+0.5)⌉;
j = -⌊(minNewVer-0.5)⌋;
i = (i > j) ? i+i+1 : j+j+1;
double scaleFactor = normalizedHeight/double(i);
i = ⌈(maxNewHor)⌉;
j = -⌊(minNewHor)⌋;
i = (i > j) ? i+i+1 : j+j+1;
int normalizedWidth = ⌈(scaleFactor * i)⌉;
// make the output width odd
if (normalizedWidth%2 == 0) normalizedWidth++;


// The normalized bitmap has dimensions normalizedHeight
// and normalizedWidth.


// Fill outputBitmap by back-projection
int shiftRow = (normalizedHeight-1)/2;
int shiftCol = (normalizedWidth-1)/2;
int intv, inth;
scaleFactor = 1.0/scaleFactor;
eigvec = eigvec * scaleFactor;
```

```
for (i = 0; i < normalizedHeight; i++) {
  for (j = 0; j < normalizedWidth; j++) {
    /* Find out what pixel position [i, j] in the
      outputBitmap maps to in the inputBitmap
      co-ordinate system (tmph, tmpv) */


    tmpv = (i - shiftRow) * eigvec[0][0] +
       (j - shiftCol) * eigvec[0][1] + mean[0];


    tmph = (i - shiftRow) * eigvec[1][0] +
       (j - shiftCol) * eigvec[1][1] + mean[1];
    // tmph and tmpv are in sub-pixel accurate positions
    intv = int(tmpv); // full pixel resolution
    inth = int(tmph);
      if (coordinate (intv, inth) and its neighbors are inside inputBitmap
boundary) {
              tmp = bilinearInterpolate(inputBitmap, tmph, tmpv);
              if (tmp > 64) outputBitmap[i][j] = 255;
    } // otherwise, 0
  }
 }
```

In addition to the following queries served by the simple "Coarse Shape (Binding Box) Representation", the normalized bitmap representation of a visual object helps in serving the query "which visual objects have a shape resembling this shape". Queries served by both Coarse Shape Representation and the Normalized bitmap representations:

1. Find the visual objects that have a particular aspect ratio (ratio of height to width).
2. Find the visual objects that are at least x% (a given percentage) dense.
3. Find the visual objects that are at most x% (a given percentage) dense.

Once the feature vectors (i.e. normalized bitmaps) are available for each visual object in each image of the database, it is possible to perform the matching based on the queries listed above, using the search engine depicted in Fig. 7.

The user provides a query bitmap, and asks for similarly shaped visual objects from the database. For the purpose of finding similarly shaped objects, a mismatch metric "M" between two normalized bitmaps is defined as the total number of positions where the two disagree. As an example, the mismatch measure M(A,Q) between two normalized bitmaps A and Q in Fig. 8 is the number of samples in the shaded area. Note in this figure that the centers of triangles are aligned and heights of the normalized bitmaps are identical.

The search engine uses the following steps to serve this request.

1. Normalize the query bitmap.

2. Obtain two mirror versions of the normalized query bitmap.

3. Obtain 180° rotated versions of the two mirror versions, for a total of four normalized query bitmaps, say Q_i, I=0,1,2 or 3. An example of these four versions is shown in Fig. 9.

4. For each normalized bitmap A_j in the database, compute the best mismatch value with the query bitmap: $d(A_j,Q) = \min_{i=0..3}M(A_j,Q_i)$.

5. Identify the indices *j*'s with low values of $d(A_j,Q)$. These indices give the best matches for the queried shape.

Fig. 10 shows a query shape and the results from a query for similar shapes.

The user provides a query bitmap (draws a shape), or enters a number (the query aspect ratio) and asks for visual objects with similar aspect ratio. The search engine uses the following steps to serve this request.

1. Normalize the query bitmap Q. Compute the aspect ratio q (height divided by width) of the normalized bitmap.

2. For each normalized bitmap A_j in the database, compute the aspect ratio a_j and the absolute difference between q and a_j, $d_j = |a_j - q|$.

3. Identify the indices *j*'s with low values of d_j. These indices give the best matches for the queried aspect ratio.

The user provides a query bitmap and asks for visual objects with similar density of valid samples. The density is the fractional number of valid samples in a bitmap. The search engine uses the following steps to serve this request.

1. Given a query bitmap Q, compute the density of valid samples, "q".
2. For each normalized bitmap A_j in the database, compute the density a_j and the absolute difference between q and a_j, $d_j = |a_j - q|$.
3. Identify the indices j's with low values of d_j. These indices give the best matches for the queried density of valid samples.

Thus the present invention provides a normalized bitmap representation of a visual object's shape for search/query/filtering applications estimating the mean and covariance of the positions of valid samples in an unnormalized bitmap, and computing the bitmap from the covariance matrix.